STAT 4410/8416 Homework 2

ISLAM MD TAHIDUL

Due on October 10, 2023

1. The data set `airquality` contains the daily air quality measurement in New York from May to Sep 1973.

```
aq <- airquality
```

Now answer the following questions: a. Compute the average temperature (`Temp`) for each month. Attach the result to the data frame `aq` as a new column called `monthly_ave_temp`. Demonstrate your results by showing the head of `aq`. **Answer:**

```
#install.packages("dplyr")
# Loading the dplyr package
library(dplyr)

# Calculate the average temperature for each month and attach it to the data frame
aq <- aq %>%
  group_by(Month) %>%
  mutate(monthly_ave_temp = mean(Temp, na.rm = T))
# Show the head of the updated data frame
head(aq)
```

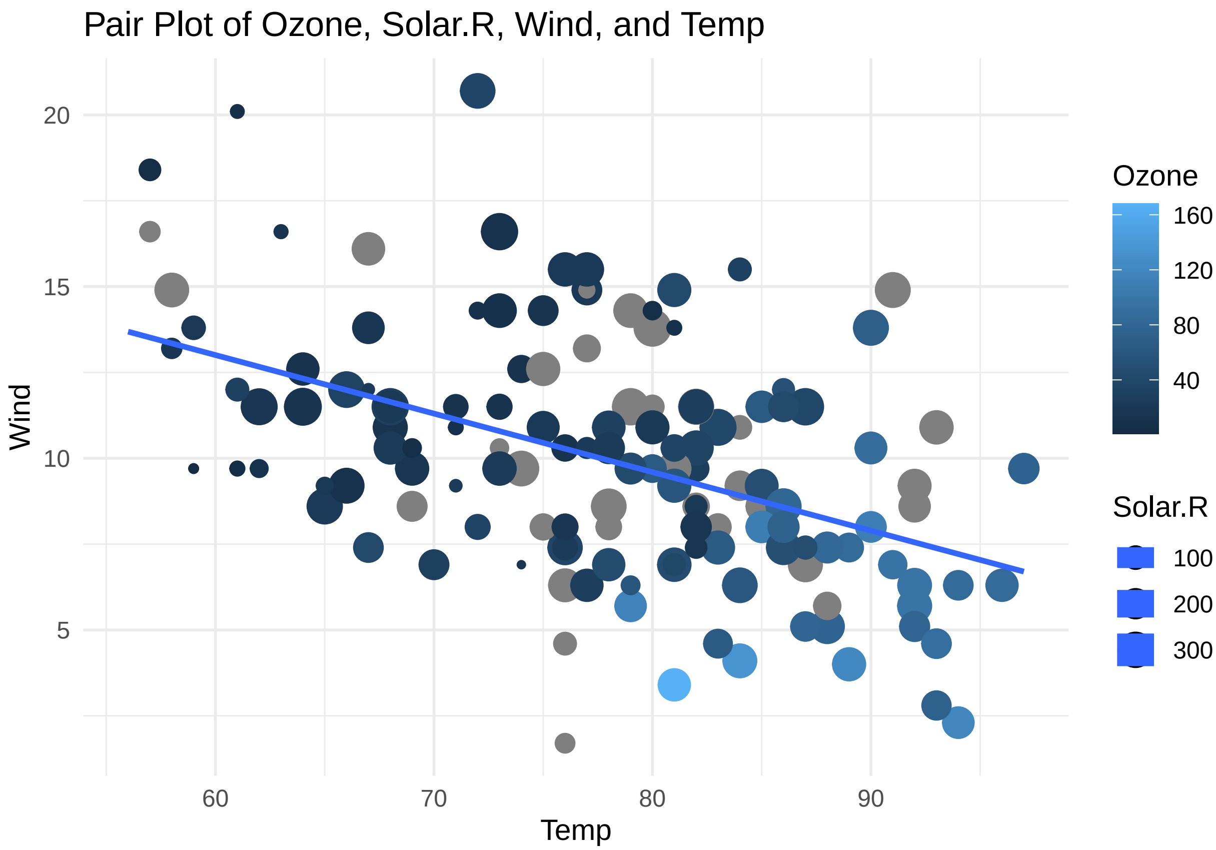
```
## # A tibble: 6 x 7
## # Groups:   Month [1]
##   Ozone Solar.R  Wind  Temp Month   Day monthly_ave_temp
##   <int>   <int> <dbl> <int> <int> <int>            <dbl>
## 1    41     190   7.4    67     5     1             65.5
## 2    36     118   8      72     5     2             65.5
## 3    12     149  12.6    74     5     3             65.5
## 4    18     313  11.5    62     5     4             65.5
## 5    NA      NA  14.3    56     5     5             65.5
## 6    28      NA  14.9    66     5     6             65.5
```

---

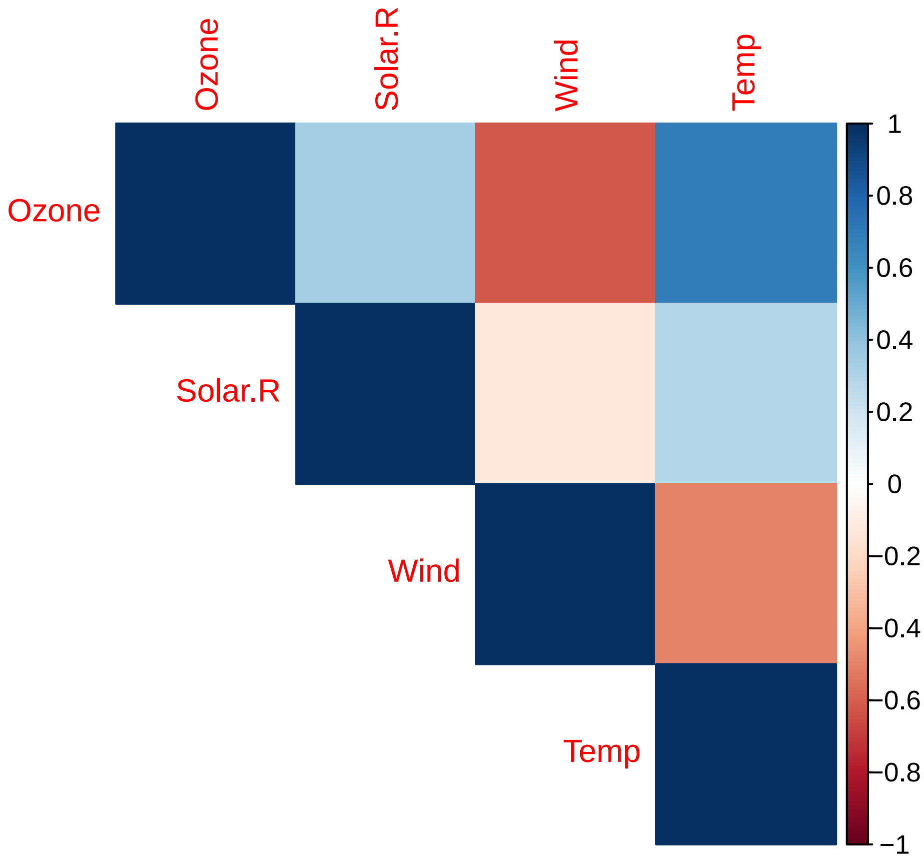
b. Draw a plot that displays the relation between the first four variables (`Ozone`, `Solar.R`, `Wind`, and `Temp`). Provide your code as well as your plot. Report highly correlated variables.

```
#install.packages("ggplot2")
library(ggplot2)

# Create a pair plot with Wind as the size aesthetic
pair_plot <- ggplot(aq, aes(x = Temp, y = Wind, color = Ozone, size =Solar.R)) +
  geom_point() +
  geom_smooth(method = "lm",se=F) +
  labs(title = "Pair Plot of Ozone, Solar.R, Wind, and Temp") +
  theme_minimal()
```

```r
# Display the pair plot
print(pair_plot)
```




```r
#install.packages("corrplot")
# Loading the necessary library for correlation plots
library(corrplot)

# Compute the correlation matrix
correlation_matrix <- cor(aq[, c("Ozone", "Solar.R", "Wind", "Temp")], use = "complete.obs")
# Create a correlation matrix plot
corrplot(correlation_matrix, method = "color", type = "upper", tl.cex = 1)
```

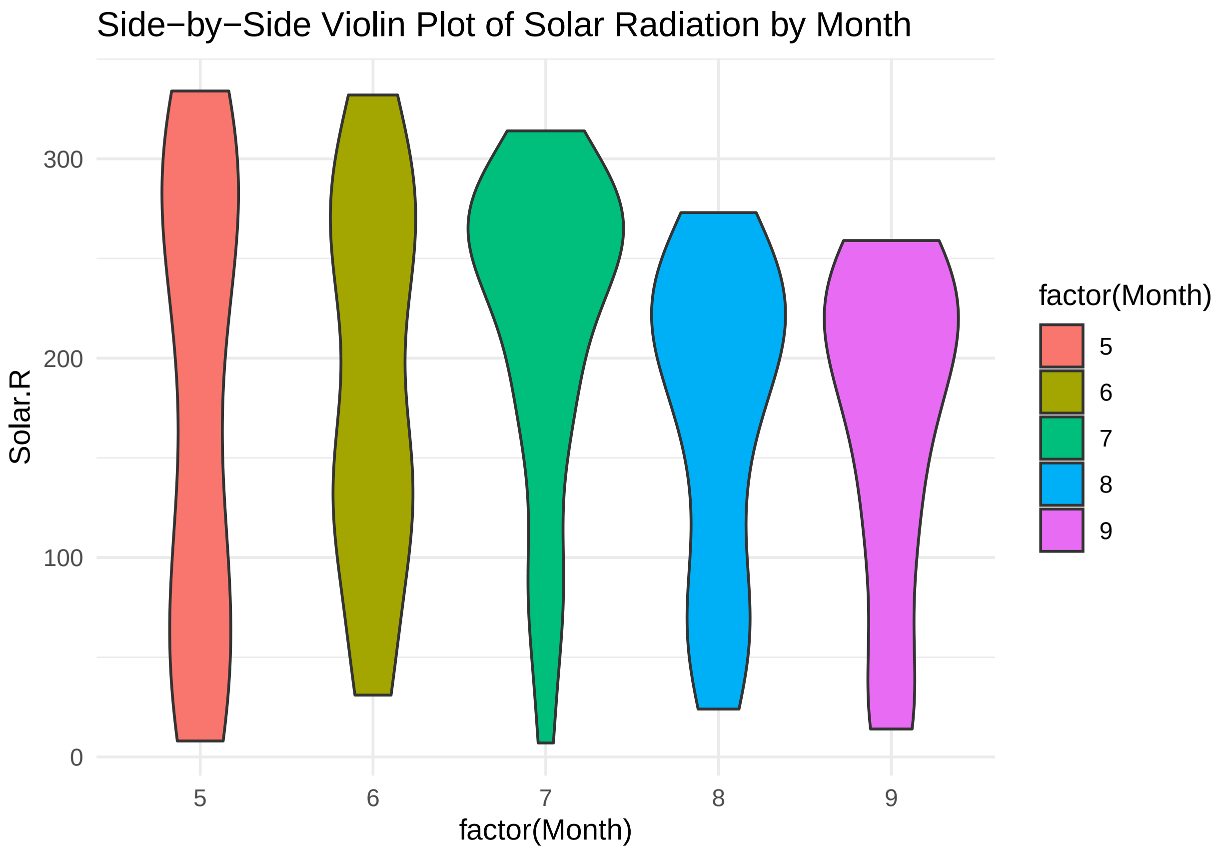


Highly positive correlated variable is `Solar.R` and `Ozone` although `Wind` and `Temp` has negative correlation.

---

c. Draw a side-by-side violin plot of solar radiation (`Solar.R`) for each month. Provide your code as well as your plot. Which month is responsible for the highest median solar radiation? How did you deal with the missing values in the `Solar.R` column?

```r
library(ggplot2)

# Filter out rows with missing values in Solar.R
aq_filtered <- aq[!is.na(aq$Solar.R), ]
# Create a violin plot
ggplot(aq_filtered, aes(x = factor(Month), y = Solar.R, fill = factor(Month))) +
  geom_violin()+
  labs(title = "Side-by-Side Violin Plot of Solar Radiation by Month") +
  theme_minimal()
```




```
# Calculate median solar radiation for each month
median_radiation <- aggregate(Solar.R ~ Month, data = aq_filtered, median)
highest_median_month <- median_radiation[which.max(median_radiation$Solar.R),"Month" ]
highest_median_month
```

```
## [1] 7
```

The variable `highest_median_month` will contain the month responsible for the highest median solar radiation. In our data which is **7** and by `[!is.na(aq$Solar.R), ]` I deal with the missing values.

---

2. We can generate an $n \times k$ matrix M and a vector V of length k for some specific values of n and k as follows:

```
set.seed(321)
n = 9
k = 5
V = sample(50, size = k, replace = TRUE)
M = matrix(rnorm(n * k), ncol = k)
```

a. Now, carefully review the following for-loop. Rewrite the code so that you perform the same job without a loop.

```
X = M
for (i in 1:n) {
  X[i, ] <- round(M[i, ] / V, digits = 4)
}
# Performing the operation without a for-loop
Y <- round(M / V, digits = 4)
```

b. Now do the same experiment for $n = 900$ and $k = 500$. Which runs faster, your code or the for-loop? Demonstrate this using the function `system.time()`.

```r
# Generate data
n <- 900
k <- 500

# Vectorized approach
system.time({
  Y <- round(M / V, digits = 4)
})
```

```
##    user  system elapsed 
##       0       0       0
```

```r
# Initialize a matrix X
X <- M
# Ensure n is not greater than the number of rows in M
n <- min(n, nrow(M))
# For-loop approach
system.time({
for (i in 1:n) {
  X[i, ] <- round(M[i, ]/V, digits = 4)
}
})
```

```
##    user  system elapsed 
##   0.001   0.000   0.001
```

as elapsed time on vectorized approach is less,so vectorized operations tend to be more efficient in `R`.

---

3. We want to generate a plot of US arrest data (USArrests). Please provide the detailed codes to answer the following questions.
    a. Obtain USA state boundary coordinates data for generating a USA map using function `map_data()` and store the data in `mdat`. Display the first few rows of data from `mdat`, noticing that there is a column called `order` that contains the true order of the coordinates.

```r
#library(ggplot2)
#install.packages("maps")
library(maps)
#US state boundary coordinates data
mdat <- map_data("state")
head(mdat)
```

```
##        long      lat group order  region subregion
## 1 -87.46201 30.38968     1     1 alabama      <NA>
## 2 -87.48493 30.37249     1     2 alabama      <NA>
## 3 -87.52503 30.37249     1     3 alabama      <NA>
## 4 -87.53076 30.33239     1     4 alabama      <NA>
## 5 -87.57087 30.32665     1     5 alabama      <NA>
## 6 -87.58806 30.32665     1     6 alabama      <NA>
```

---

b. You will find USA crime data in the data frame called `USArrests`. Standardize the crime rates and create a new column called `state` so that all state names are in lower case. Store this new data in an object called `arrest` and report the first few rows of `arrest`.

```r
# Load the USArrests dataset
data("USArrests")
glimpse(USArrests)
```

```
## Rows: 50
## Columns: 4
## $ Murder   <dbl> 13.2, 10.0, 8.1, 8.8, 9.0, 7.9, 3.3, 5.9, 15.4, 17.4, 5.3, 2.~
## $ Assault  <int> 236, 263, 294, 190, 276, 204, 110, 238, 335, 211, 46, 120, 24~
## $ UrbanPop <int> 58, 48, 80, 50, 91, 78, 77, 72, 80, 60, 83, 54, 83, 65, 57, 6~
## $ Rape     <dbl> 21.2, 44.5, 31.0, 19.5, 40.6, 38.7, 11.1, 15.8, 31.9, 25.8, 2~
```

```r
# Create a new data frame with standardized crime rates and lowercase state names
arrest <- USArrests
arrest$state <- tolower(rownames(USArrests))  # Add a 'state' column with lowercase state names
arrest[, -5] <- scale(arrest[, -5])  # Standardize the crime rate columns (Murder, Assault, UrbanPop, R

# Report the first few rows of arrest
head(arrest)
```

```
##                 Murder  Assault   UrbanPop         Rape      state
## Alabama    1.24256408 0.7828393 -0.5209066 -0.003416473    alabama
## Alaska     0.50786248 1.1068225 -1.2117642  2.484202941     alaska
## Arizona    0.07163341 1.4788032  0.9989801  1.042878388    arizona
## Arkansas   0.23234938 0.2308680 -1.0735927 -0.184916602   arkansas
## California 0.27826823 1.2628144  1.7589234  2.067820292 california
## Colorado   0.02571456 0.3988593  0.8608085  1.864967207   colorado
```

---

c. Merge the two data sets `mdat` and `arrest` by state name. Note: merging will change the order of the coordinates data. So, order the data back to the original order and store the merged-ordered data in `odat`. Report the first few rows of data from `odat`.

```r
# Merge the two data sets by state name
odat <- merge(mdat, arrest, by.x = "region", by.y = "state", all.x = TRUE)

# Reorder the data to match the original order
odat <- odat[order(odat$order), ]

# Report the first few rows of data from odat
head(odat)
```

```
##    region      long      lat group order subregion   Murder   Assault
## 1 alabama -87.46201 30.38968     1     1      <NA> 1.242564 0.7828393
## 2 alabama -87.48493 30.37249     1     2      <NA> 1.242564 0.7828393
## 6 alabama -87.52503 30.37249     1     3      <NA> 1.242564 0.7828393
## 7 alabama -87.53076 30.33239     1     4      <NA> 1.242564 0.7828393
## 8 alabama -87.57087 30.32665     1     5      <NA> 1.242564 0.7828393
## 9 alabama -87.58806 30.32665     1     6      <NA> 1.242564 0.7828393
##     UrbanPop         Rape
## 1 -0.5209066 -0.003416473
## 2 -0.5209066 -0.003416473
## 6 -0.5209066 -0.003416473
## 7 -0.5209066 -0.003416473
## 8 -0.5209066 -0.003416473
## 9 -0.5209066 -0.003416473
```

d. All the columns of `odat` are not necessary for our analysis. So, obtain a subset by selecting only the columns `long`, `lat`, `group`, `region`, `Murder`, `Assault`, `UrbanPop`, and `Rape`. Store the data in `sdat` and report the first few rows.

```
# Selecting specific columns
columns_to_select <- c("long", "lat", "group", "region", "Murder", "Assault", "UrbanPop", "Rape")
sdat <- odat[, columns_to_select]

# Report the first few rows of sdat
head(sdat)
```

```
##        long      lat group  region   Murder   Assault   UrbanPop         Rape
## 1 -87.46201 30.38968     1 alabama 1.242564 0.7828393 -0.5209066 -0.003416473
## 2 -87.48493 30.37249     1 alabama 1.242564 0.7828393 -0.5209066 -0.003416473
## 6 -87.52503 30.37249     1 alabama 1.242564 0.7828393 -0.5209066 -0.003416473
## 7 -87.53076 30.33239     1 alabama 1.242564 0.7828393 -0.5209066 -0.003416473
## 8 -87.57087 30.32665     1 alabama 1.242564 0.7828393 -0.5209066 -0.003416473
## 9 -87.58806 30.32665     1 alabama 1.242564 0.7828393 -0.5209066 -0.003416473
```

e. Melt the data frame `sdat` with id variables `long`, `lat`, `group`, `region`. Store the molten data in `msdat` and report the first few rows of data. Use two ways to do this (`reshape2` and `tidyr`).

```
#Tidyr way
# Load the tidyr library if not already loaded
library(tidyr)

# Melt the data frame using pivot_longer() from tidyr
msdat <- pivot_longer(sdat, cols = Murder:Rape, names_to = "Crime", values_to = "Rate")

# Report the first few rows of msdat
head(msdat)
```

```
## # A tibble: 6 x 6
##    long   lat group region  Crime        Rate
##   <dbl> <dbl> <dbl> <chr>   <chr>       <dbl>
## 1 -87.5  30.4     1 alabama Murder    1.24
## 2 -87.5  30.4     1 alabama Assault   0.783
## 3 -87.5  30.4     1 alabama UrbanPop -0.521
## 4 -87.5  30.4     1 alabama Rape     -0.00342
## 5 -87.5  30.4     1 alabama Murder    1.24
## 6 -87.5  30.4     1 alabama Assault   0.783
```

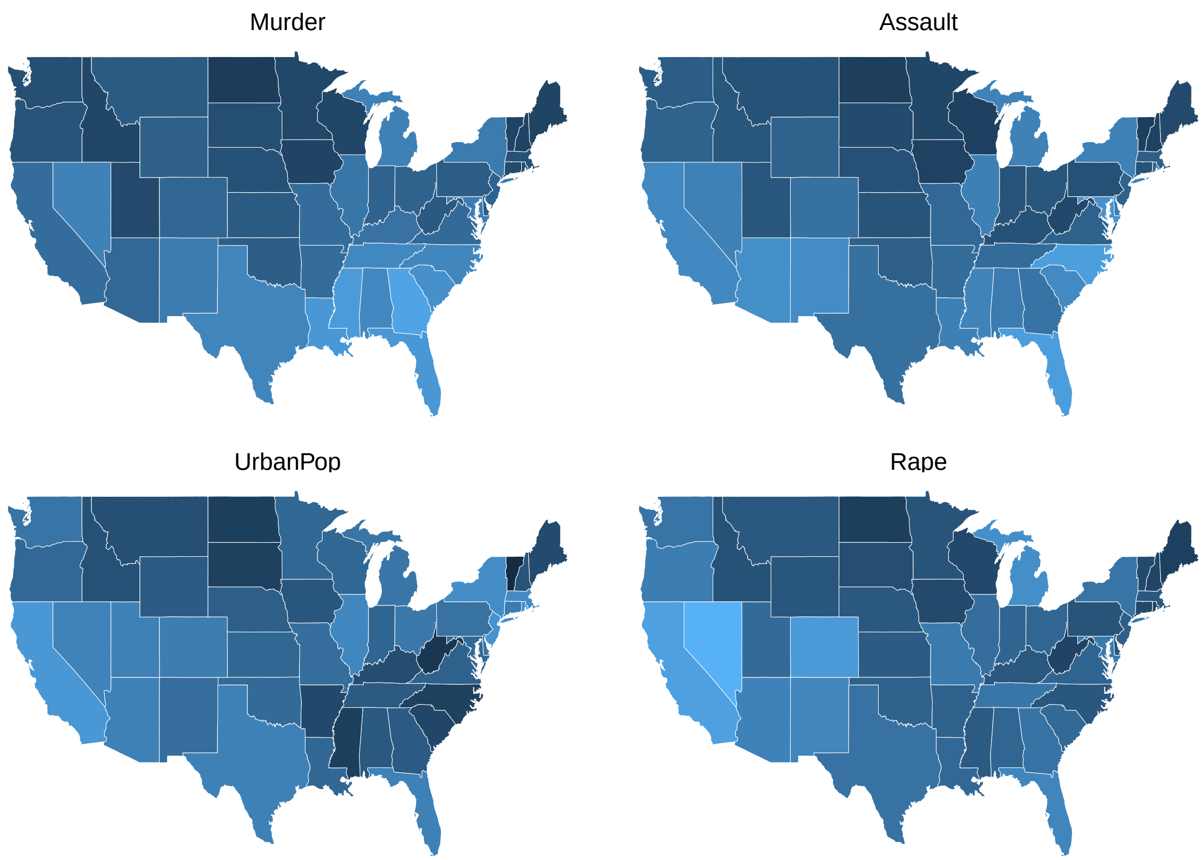
```
#-------------------------------------------
#reshape2 way
# Loading the reshape2 library
library(reshape2)

# Melt the data frame using melt() from reshape2
msdat <- melt(sdat, id.vars = c("long", "lat", "group", "region"))

# Report the first few rows of msdat
head(msdat)
```

```
##        long      lat group  region variable    value
## 1 -87.46201 30.38968     1 alabama   Murder 1.242564
## 2 -87.48493 30.37249     1 alabama   Murder 1.242564
```

```
## 3 -87.52503 30.37249      1 alabama   Murder 1.242564
## 4 -87.53076 30.33239      1 alabama   Murder 1.242564
## 5 -87.57087 30.32665      1 alabama   Murder 1.242564
## 6 -87.58806 30.32665      1 alabama   Murder 1.242564
```

f. The molten data frame `msdat` is now ready to be plotted. Let's use `msdat` from the `reshape2` output. Create a plot showing the USA state map, fill the color by `value`, and `facet_wrap` with `variable`. Please don't add any legend and make sure that facetting labels are identified so that we can compare the facetted plots.

```r
library(ggplot2)

# Create the plot
ggplot(msdat, aes(x = long, y = lat, group = group, fill = value)) +
  geom_polygon(color="white",size=0.1) +
  facet_wrap(~ variable, ncol = 2) +
  theme_void() +
  theme(legend.position = "none")
```



g. Now examine the plot you have generated in question (f) and answer the following questions based on what you see in the plot.
   i. For each crime, name two states with its highest rate.

```r
library(dplyr)

# Create a data frame for each crime variable
murder_summary <- msdat %>%
  filter(variable == "Murder") %>%  # Filter the "Murder" variable
```

```r
  group_by(region) %>%
  summarize(total_murder = sum(value)) %>%
  arrange(desc(total_murder))  # Arrange in descending order of total murder

assault_summary <- msdat %>%
  filter(variable == "Assault") %>%  # Filter the "Assault" variable
  group_by(region) %>%
  summarize(total_assault = sum(value)) %>%
  arrange(desc(total_assault))  # Arrange in descending order of total assault

urbanpop_summary <- msdat %>%
  filter(variable == "UrbanPop") %>%  # Filter the "UrbanPop" variable
  group_by(region) %>%
  summarize(total_urbanpop = sum(value)) %>%
  arrange(desc(total_urbanpop))  # Arrange in descending order of total UrbanPop

rape_summary <- msdat %>%
  filter(variable == "Rape") %>%  # Filter the "Rape" variable
  group_by(region) %>%
  summarize(total_rape = sum(value)) %>%
  arrange(desc(total_rape))  # Arrange in descending order of total Rape

# Combine the results into a single data frame
crime_results <- bind_rows(
  data.frame(Crime = "Murder", Highest_Region = murder_summary$region[1], Second_Highest_Region = murde
  data.frame(Crime = "Assault", Highest_Region = assault_summary$region[1], Second_Highest_Region = assa
  data.frame(Crime = "UrbanPop", Highest_Region = urbanpop_summary$region[1], Second_Highest_Region = ur
  data.frame(Crime = "Rape", Highest_Region = rape_summary$region[1], Second_Highest_Region = rape_summa
)
#For this code i seek help from google
# Print the combined results
print(crime_results)
```

```
##      Crime Highest_Region Second_Highest_Region
## 1   Murder        florida                 texas
## 2  Assault        florida        north carolina
## 3 UrbanPop          texas            california
## 4     Rape       michigan            california
```

so, by the output of the `crime_results` we can conclude that:
- Florida and Texas rank highest in terms of murder rates. - Florida and then North Carolina have most assault rates. - Michigan and California have reported higher incidences of rape.

ii. Do you think a larger urban population is indicative of a higher murder rate? Why or why not?

```r
crime_results
```

```
##      Crime Highest_Region Second_Highest_Region
## 1   Murder        florida                 texas
## 2  Assault        florida        north carolina
## 3 UrbanPop          texas            california
## 4     Rape       michigan            california
```

Yes.As Texas state is number one in urban population and second on most murder cases ,so i can conclude that higher population may indicate a chances of higher percentage of murder.

h. In question (b) we standardized the crime rates. Why do you think we did this? Explain what would happen if we did not standardize the data. **Answer:** -Unit Consistency: Standardization ensures that all variables are on the same scale. In the original data, Murder might be measured in homicides per 100,000 people, Assault in the number of aggravated assaults per 100,00 people.These units are different, making it challenging to compare and analyze them effectively. -Mitigating Skewness and Outliers: Standardization can help mitigate the effects of skewed distributions and outliers. It reduces the impact of extreme values and makes the data more normally distributed.

what if we did not standardize the data: -The variables would remain on different scales. -Outliers or extreme values in one variable could disproportionately influence statistical analyses -In machine learning and statistical modeling, standardization is often a crucial preprocessing step

i. In question (c) we ordered the data after merging. Why do you think we had to do this? Explain what would happen if we did not. **Answer:** Reordering the data after merging ensures that the merged dataset accurately represents the original geographic coordinates and preserves any meaningful relationships in the data. It is a crucial step to maintain data integrity and ensure correct interpretations and analysis. Failure to do so resulting in a misaligned map or visualization.

4. Life expectancy data for four countries can be obtained from the world bank database found at github. It contains life expectancy in years for different genders. Now answer the following questions.
   a. Read the data from the above link and display the first few rows of data.

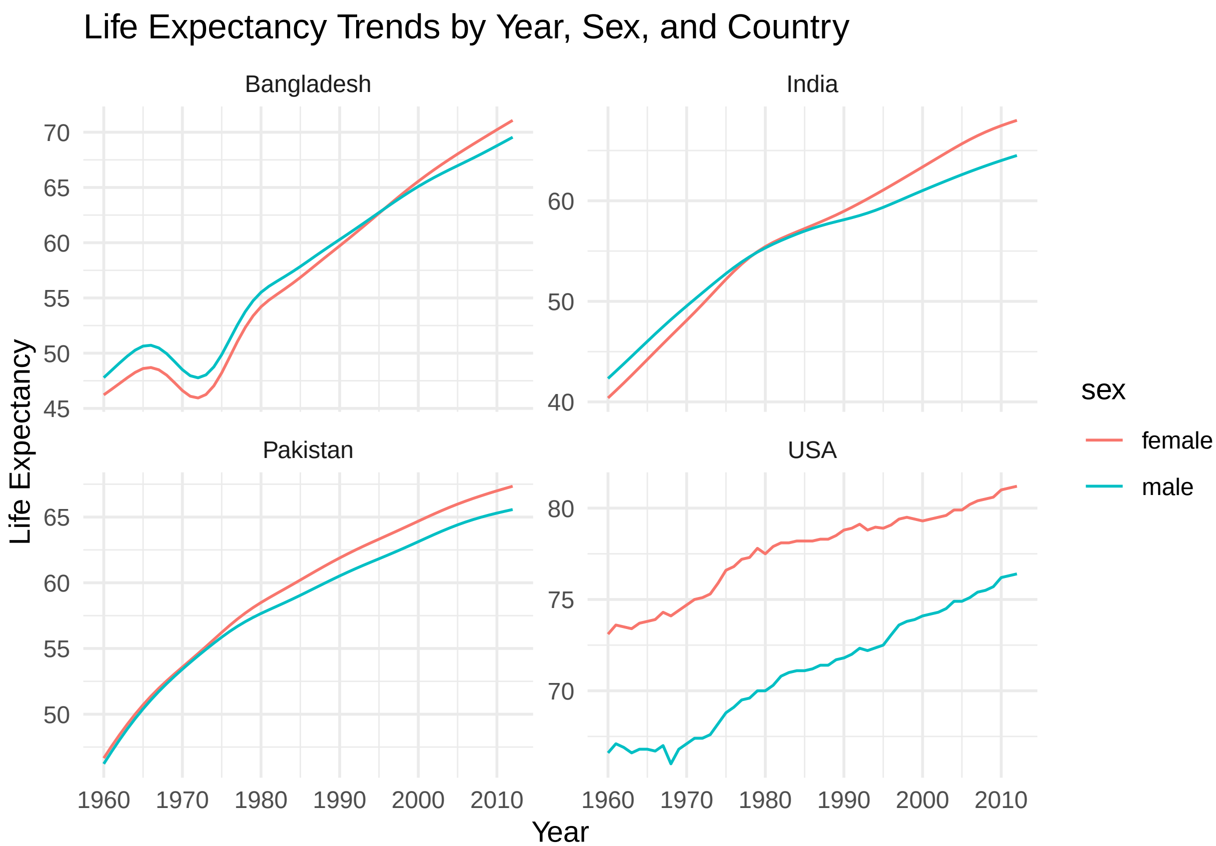
```
# URL of the CSV file
url <- "http://mamajumder.github.io/data-science/data/life-expectancy.csv"

# Read the data from the URL
life_expectancy_data <- read.csv(url)

# Display the first few rows of the data
head(life_expectancy_data)
```

```
##   year    sex Bangladesh  India Pakistan  USA
## 1 1960 female     46.224 40.391   46.655 73.1
## 2 1960   male     47.787 42.329   46.223 66.6
## 3 1961 female     46.731 41.125   47.564 73.6
## 4 1961   male     48.445 43.052   47.156 67.1
## 5 1962 female     47.254 41.876   48.426 73.5
## 6 1962   male     49.104 43.784   48.044 66.9
```

b. Generate a plot showing trend lines of life expectancy by year. Color them by sex and facet by country. Include your code with the plot.

```
# Load the necessary libraries
library(ggplot2)
library(dplyr)

# Create the plot
life_expectancy_plot <- life_expectancy_data %>%
  pivot_longer(cols = -c(year, sex), names_to = "Country", values_to = "LifeExpectancy") %>%
  ggplot(aes(x = year, y = LifeExpectancy, color = sex)) +
  geom_line() +
  facet_wrap(~ Country, scales = "free_y", ncol = 2) +
  labs(
    title = "Life Expectancy Trends by Year, Sex, and Country",
    x = "Year",
    y = "Life Expectancy"
  ) +
```

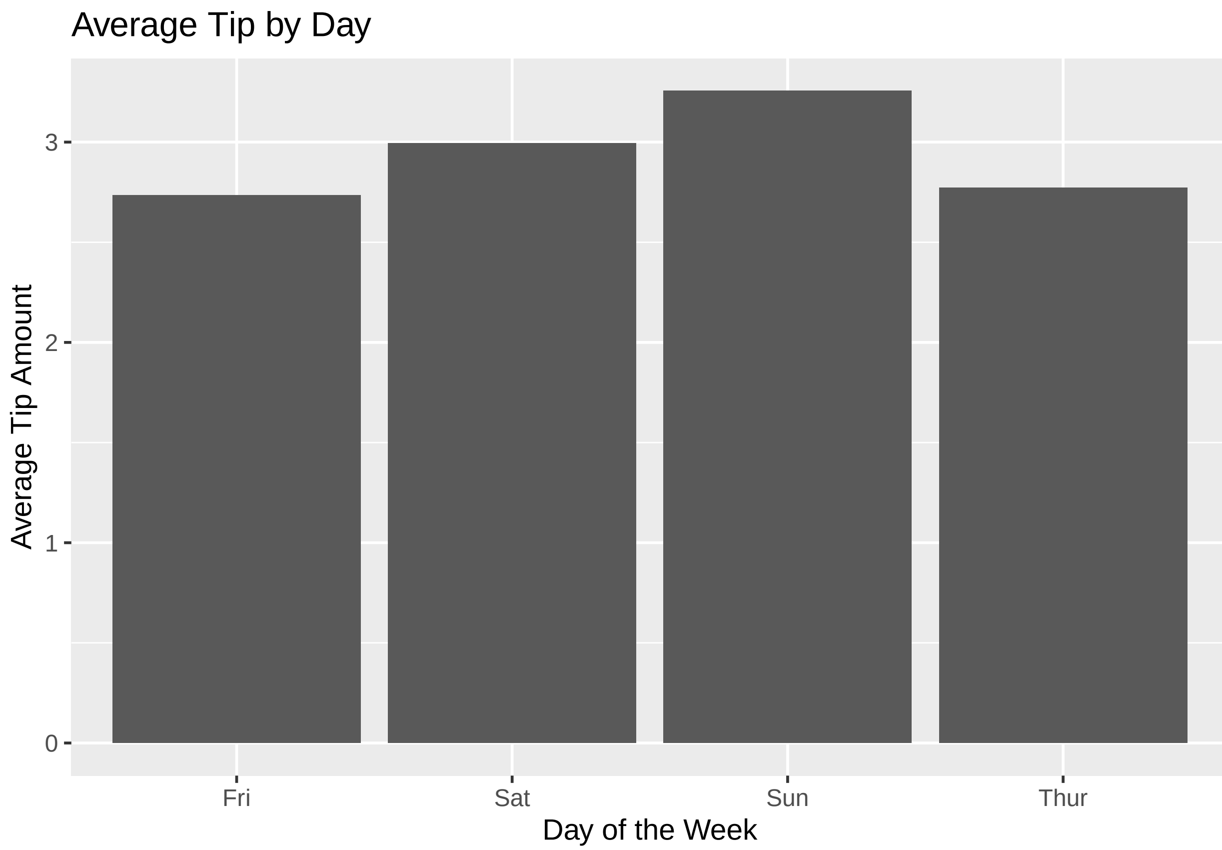
```r
  theme_minimal()

# Display the plot
print(life_expectancy_plot)
```



c. Explain what interesting features you noticed in the plot you made in question (b). **Answer:** 1.The United States shows the highest life expectancy compared to other nations. 2.Females have a greater life expectancy than males. 3.There is a drastic drop in the life expectancy graph for Bangladesh in early 1970s. 4.From 1960s, in India and Bangladesh, male life expectancy exceeded by female.later this trend reversed. 5.Most noticeable is that over the years life expectancy is increased a lot for all of the countries.

5. For the following questions please use the data frame `tips`.
   a. Create a bar chart that shows the average tip by day.

```r
# Load the ggplot2 library
library(ggplot2)
# Create the bar chart
ggplot(tips, aes(x = day, y = tip)) +
  geom_bar(stat = "summary", fun = "mean") +
  labs(title = "Average Tip by Day", x = "Day of the Week", y = "Average Tip Amount")
```



b. Compute the average tip, total tip, and average size grouped by smoker and day. i.e., For each combination of smoker and day you should have a row of these summaries. Report these results in a nice table.

```r
# Load the necessary libraries
library(dplyr)
library(knitr)

# Group the tips dataset by smoker and day, and compute the required summaries
summary_table <- tips %>%
  group_by(smoker, day) %>%
  summarise(
    Average_Tip = mean(tip),
    Total_Tip = sum(tip),
    Average_Size = mean(size)
  )

# Use kable to create a nice table
summary_table %>%
  kable(caption = "Summary of Tips and Table Size by Smoker and Day")
```

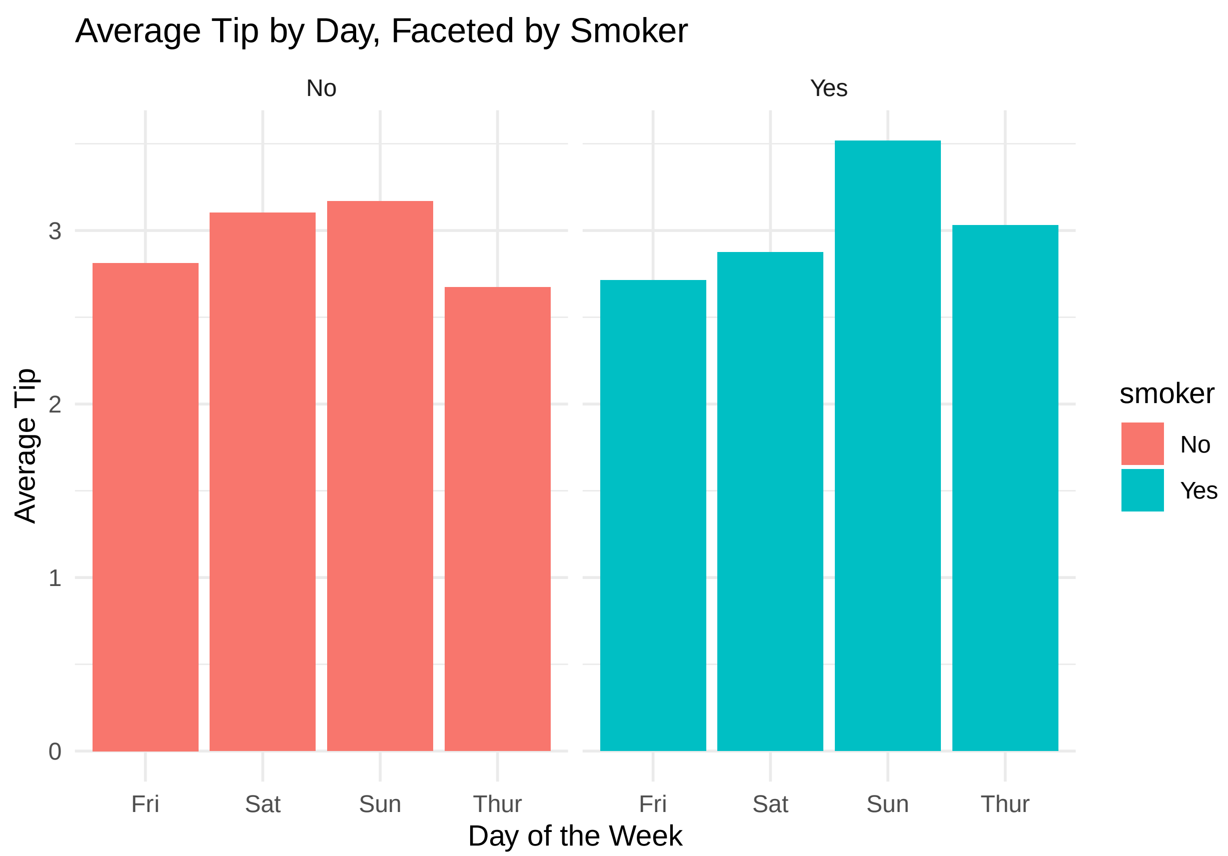
Table 1: Summary of Tips and Table Size by Smoker and Day

smoker	day	Average_Tip	Total_Tip	Average_Size
No	Fri	2.812500	11.25	2.250000
No	Sat	3.102889	139.63	2.555556
No	Sun	3.167895	180.57	2.929825

smoker	day	Average_Tip	Total_Tip	Average_Size
No	Thur	2.673778	120.32	2.488889
Yes	Fri	2.714000	40.71	2.066667
Yes	Sat	2.875476	120.77	2.476190
Yes	Sun	3.516842	66.82	2.578947
Yes	Thur	3.030000	51.51	2.352941

c. Create a bar chart that shows average tip by day, faceted by smoker.

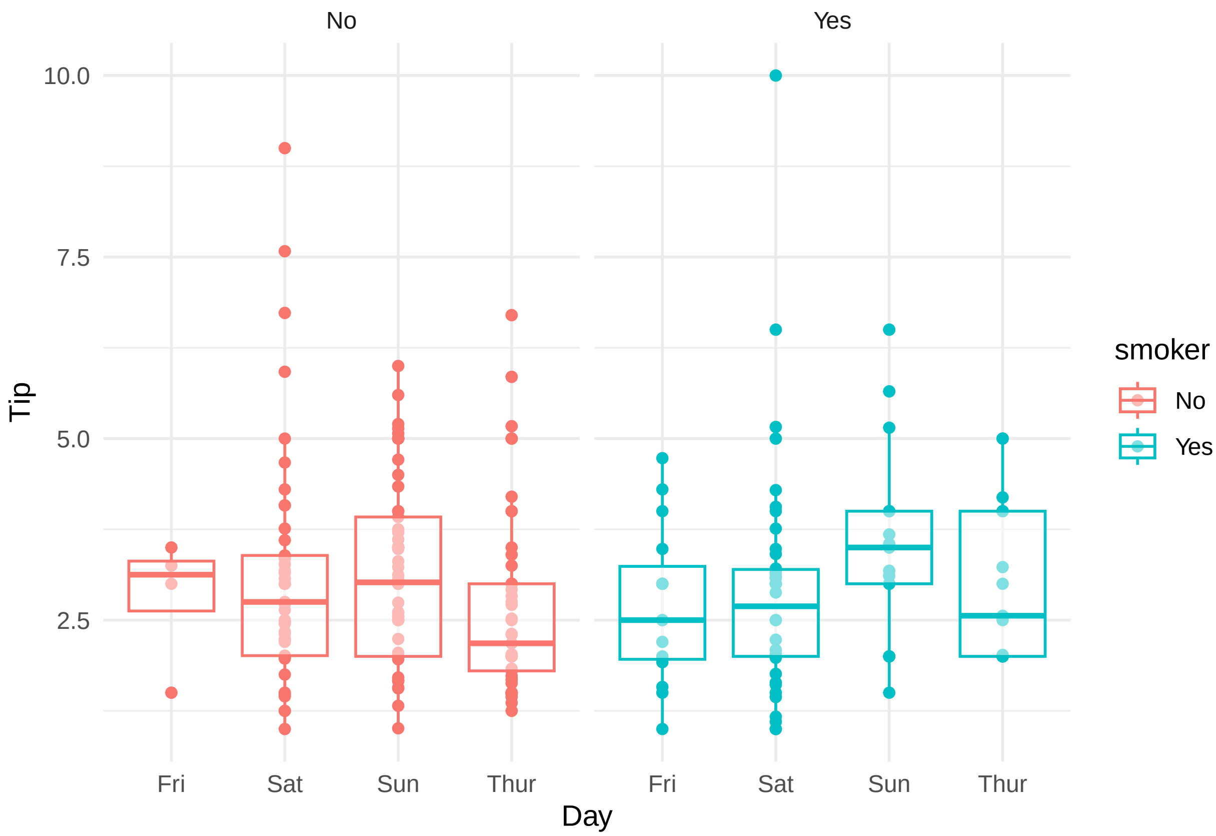
```
# Load the ggplot2 library
library(ggplot2)

# Create the bar chart
ggplot(tips, aes(x = day, y = tip, fill = smoker)) +
  geom_bar(stat = "summary", fun = "mean", position = "dodge") +
  labs(title = "Average Tip by Day, Faceted by Smoker",
       x = "Day of the Week",
       y = "Average Tip") +
  facet_wrap(~smoker) +
  theme_minimal()
```



d. In questions (a) and (c), we plotted a summary of our data which does not show us the whole picture. In practice, we would like to see all of the data. What plot do you suggest would serve a similar purpose to the one in question (c)? In other words, what would be a better plot to show than tips by day, facetted by smoker? Please produce this plot and include your code.

Answer: In my opinion Box plot would be a great choice.Because box plots are used to visualize the distribution of a dataset by displaying key summary statistics, including the median, quartiles, and potential outliers.To compare tips by day with facetted by smoker it will be allowing for quick visual comparisons.

```
# Load the ggplot2 library
library(ggplot2)

# Create the plot with modified facet appearance
ggplot(tips, aes(day, tip, color = smoker)) +
  geom_point() +
  geom_boxplot(alpha = 1/2) +
  facet_wrap(~smoker, nrow = 1) +  # Display facets in a single row
  xlab("Day") +
  ylab("Tip") +
  theme_minimal()
```



6. We have the following data set:

```
myDat = read.csv("http://mamajumder.github.io/data-science/data/reshape-source.csv")
kable(myDat)
```

player	track	walking	cycling
1	A	408	43
1	B	402	31
1	C	386	41
2	A	373	53
2	B	404	41
2	C	422	30
3	A	403	25
3	B	393	46

player	track	walking	cycling
3	C	422	48

We want to reshape the data and produce the following output:

player	variable	A	B	C
1	walking	408	402	386
1	cycling	43	31	41
2	walking	373	404	422
2	cycling	53	41	30
3	walking	403	393	422
3	cycling	25	46	48

Provide code that will produce this desired output. Demonstrate your answer by displaying the output as well.

```
result <- myDat %>%
  pivot_longer(cols = c(walking, cycling), names_to = "variable", values_to = "value") %>%
  pivot_wider(names_from = track, values_from = value)
# Print the result
result
```

```
## # A tibble: 6 x 5
##   player variable     A     B     C
##    <int> <chr>    <int> <int> <int>
## 1      1 walking    408   402   386
## 2      1 cycling     43    31    41
## 3      2 walking    373   404   422
## 4      2 cycling     53    41    30
## 5      3 walking    403   393   422
## 6      3 cycling     25    46    48
```

7. **Ordering the factor** In class, we have seen how to order factors. Suppose we have the following data about a certain value obtained during particular months of the year;

```
month = c("July", "June", "September", "May", "October", "August")
value = c(35, 72, 14, 23, 60, 105)
df = data.frame(month, value)
```

Now please do the following:

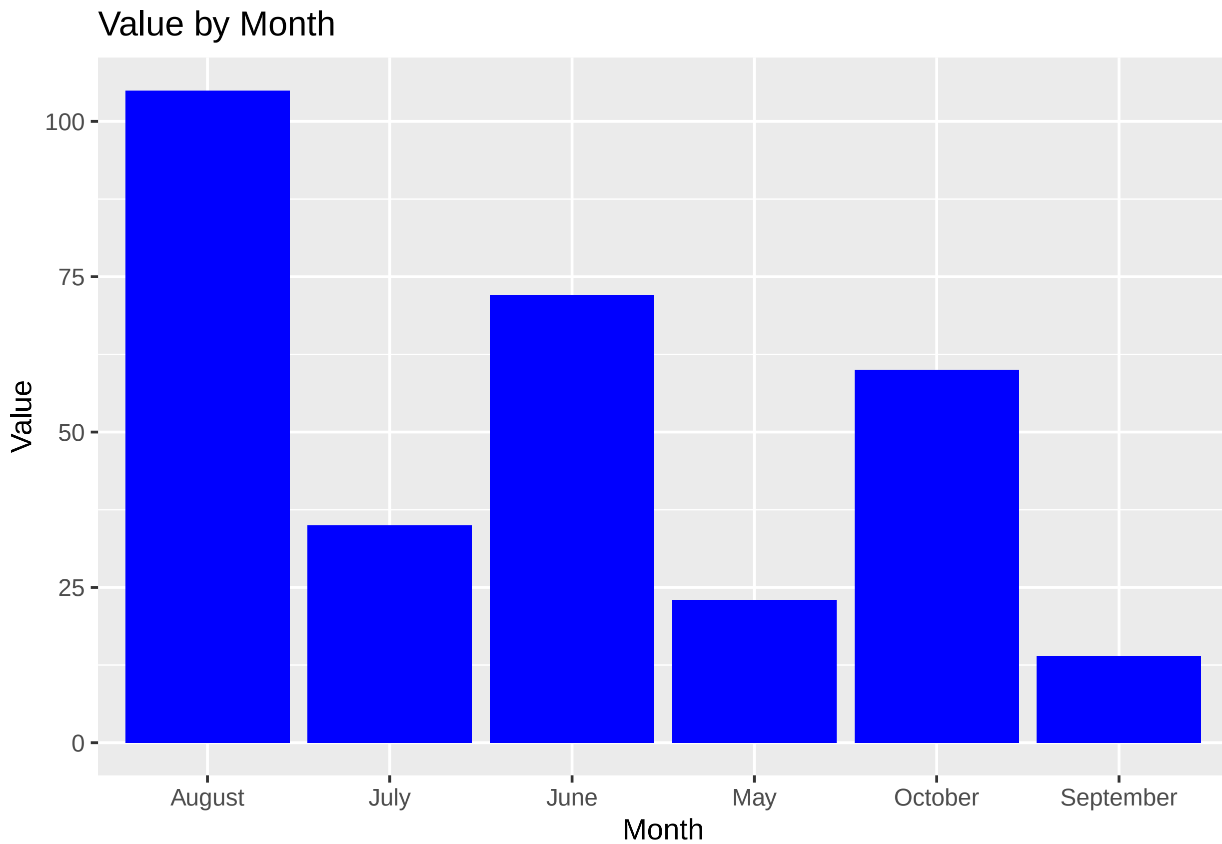
a. Convert the month column of the data frame `df` into a factor column. Demonstrate that it is indeed converted into a factor column.

```
# Convert the "month" column into a factor
df$month <- as.factor(df$month)

# Check the data type of the "month" column
class(df$month)
```

```
## [1] "factor"
```

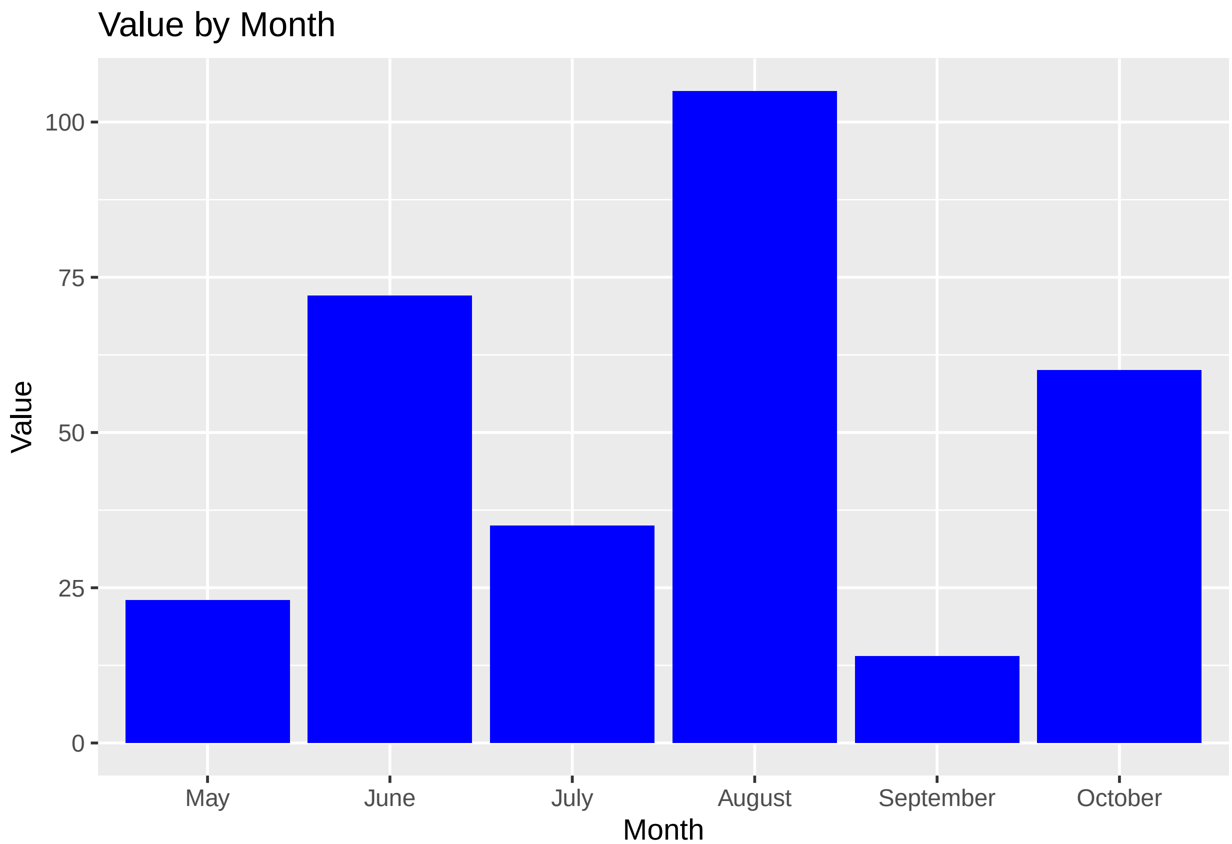
b. Now generate a bar chart showing the value for different months.

```r
ggplot(df, aes(x = month, y = value)) +
  geom_bar(stat = "identity", fill = "blue") +
  labs(title = "Value by Month", x = "Month", y = "Value")
```



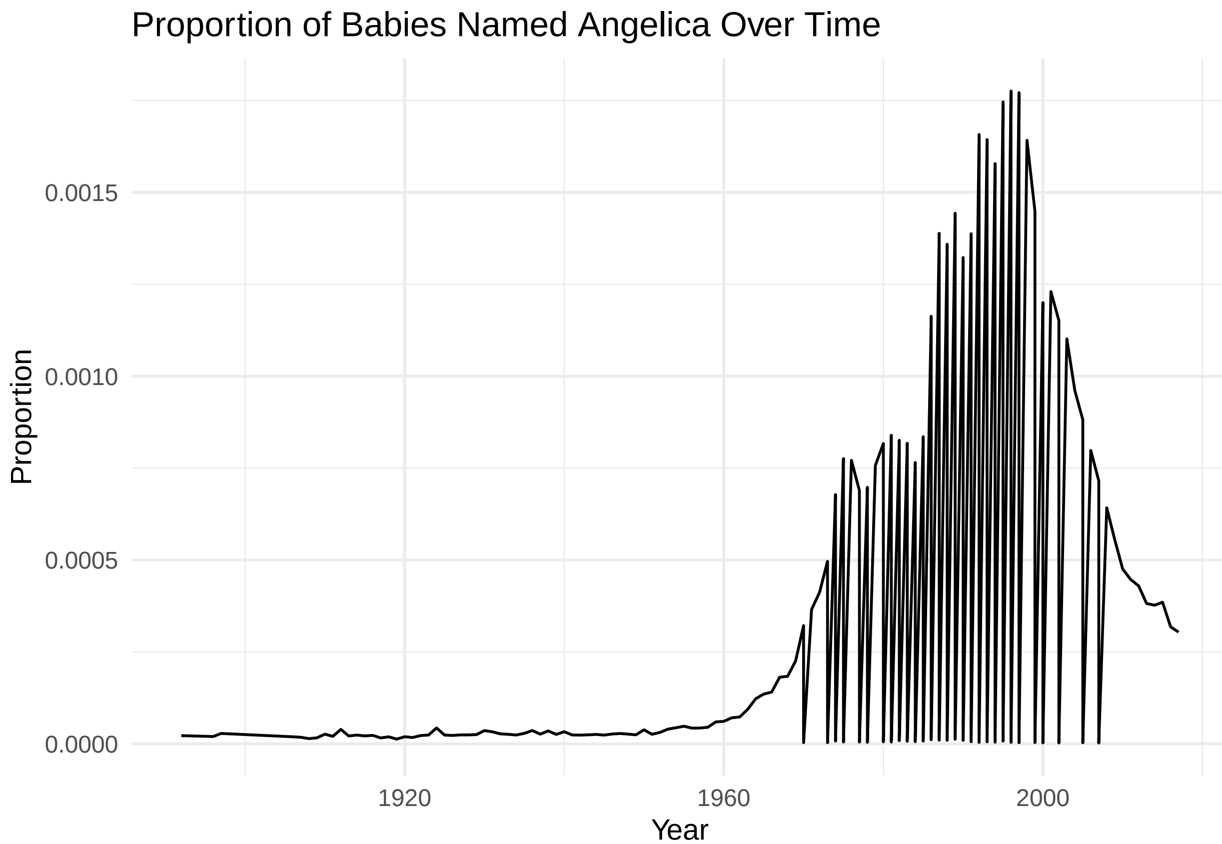
c. Notice the order of the levels of the months is not natural, instead the plot shows the dictionary order. Now, order the bars according to the natural order of the levels of the class (months of the year as they appear in chronological order) and regenerate the bar graph.

```r
# Define the desired order of months
desired_order <- c("May", "June", "July", "August", "September", "October")

# Set the "month" column as a factor with desired order
df$month <- factor(df$month, levels = desired_order)

# Create the bar chart with ordered months
ggplot(df, aes(x = month, y = value)) +
  geom_bar(stat = "identity", fill = "blue") +
  labs(title = "Value by Month", x = "Month", y = "Value")
```



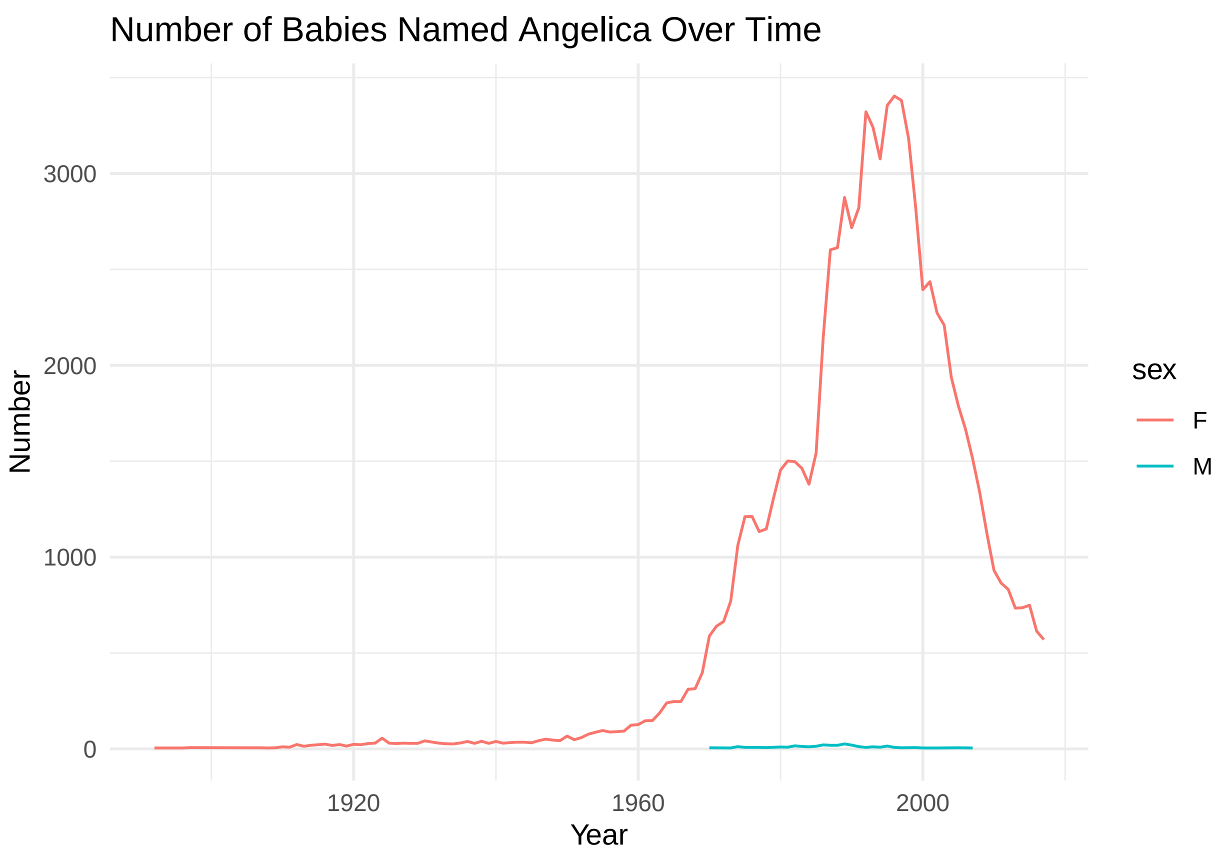
8. Install the `babynames` package with `install.packages()`. This package includes data from the Social Security Administration about American baby names over a wide range of years. Generate a plot of the reported proportion of babies born with the name `Angelica` over time. Do you notice anything odd about the plotted data? (Hint: you should.) If so, describe the issue and generate a new plot that adjusts for this problem. Make sure you show both plots along with all code that was used to generate them.

```r
#install.packages("babynames")
library(babynames)

# Filter the data for the name 'Angelica'
angelica_data <- babynames %>%
  filter(name == "Angelica")
# Plot the proportion of babies with the name 'Angelica' over time
ggplot(angelica_data, aes(x = year, y = prop)) +
  geom_line() +
  labs(title = "Proportion of Babies Named Angelica Over Time",
       x = "Year",
       y = "Proportion") +
  theme_minimal()
```



if we generate a plot based on proportion,then its hard to read from the plot.As number of name column (n) available,so in my opinion it is best to generate a plot based on number and draw lines based on sex.

```
ggplot(babynames %>% filter(name == "Angelica"), aes(x = year, y = n)) +
  geom_line(aes(color = sex)) +
  labs(title = "Number of Babies Named Angelica Over Time",
       x = "Year",
       y = "Number") +
  theme_minimal()
```



9. Suppose we have a vector of data as follows:

```
myVector = c(-15, -10, -5, 0, 5, 10, 15, 20)
```

```
a.  Using the function `tapply()`, separately compute the means of the first three values, next two valu
```

```
means <- tapply(myVector, INDEX = rep(1:3, c(3, 2, 3)), FUN = mean)

# Print the means
print(means)
```

```
##     1     2     3
## -10.0   2.5  15.0
```

```
b.  Now repeat question (a), but instead of computing means, you will compute the sum of squares. Again
```

```
# Compute the sum of squares of the specified subsets
sum_of_squares <- tapply(myVector^2, INDEX = rep(1:3, c(3, 2, 3)), FUN = sum)

# Print the sum of squares
print(sum_of_squares)
```

```
##   1   2   3
## 350  25 725
```